

Practical Application of AI and Large Language Models in Software Engineering Education

Vasil Kozov¹, Galina Ivanova², Desislava Atanasova³

Dept. of Informatics, University of Ruse "Angel Kanchev", Ruse, Bulgaria^{1,3}

Dept of Computer Systems and Technology, University of Ruse "Angel Kanchev", Ruse, Bulgaria²

Abstract—Subjects with limited application in the software industry like AI have recently received tremendous boon due to the development and raise of publicity of LLMs. LLM-powered software has a wide array of practical applications that must be taught to Software Engineering students, so that they can be relevant in the field. The speed of technological change is extremely fast, and university curriculums must include those changes. Renewing and creating new methodologies and workshops is a difficult task to complete successfully in such a dynamic environment full of cutting-edge technologies. This paper aims to showcase our approach to using LLM-powered software for AI generated images, like Stable diffusion and code generation tools like ChatGPT in workshops for two relevant subjects – Analysis of Software Requirements and Specifications, as well as Artificial Intelligence. A comparison between the different available LLMs that generate images is made, and the choice between them is explained. Student feedback is shown and a general positive and motivational impact is noted during and after the workshop. A brief introduction that covers the subjects where AI is applied is made. The proposed solutions for several uses of AI in the field of higher education, more specifically software engineering, are presented. Several workshops have been made and included in the curriculum. The results of their application have been noted and an analysis is made. More propositions on further development based on the gained experience, feedback and retrieved data are made. Conclusions are made on the application of AI in higher education and different ways to utilize such tools are presented.

Keywords—Application of AI-powered software; AI generated images; software engineering; stable diffusion; higher education

I. INTRODUCTION

With the release of popular and easy to use text based large language models, Artificial intelligence (AI) chatbots and image generation AI, it has become a necessity to teach students how to properly use them for the correct purposes. A big detriment of using AI by first and second year students is their lack of understanding in the subjects where they are trying to use AI. This sadly leads to less problem-solving thinking and doing actual effort to work on tasks. This in turn leads to less brain development and lack of practical skills. Teaching students how to correctly use such tools to enhance their learning is imperative for developing them into good specialists that may in turn use AI powered software and hardware to their benefit.

The Software Engineering bachelor's degree was chosen for the inclusion of teaching AI. Based on our experience, the more technically advanced and tech savvy students are more

receptive towards the inclusion of new and experimental approaches. This was the main factor in our choice of who to use this approach with. Several subjects were selected, where the methodologies were applied. Workshops and materials to help with training the students were created. In the current report the focus will be on the application of Large Language Models (LLMs) in two subjects - Artificial intelligence (6th semester) and Analyzing system requirements and specifications (5th semester). It will also be discussed how to use LLMs to enhance the subject Introduction to Programming by writing unit tests (while that is still in the testing phase and has only been partially applied), as well as future ideas for the inclusion of image generation AI in Computer Graphics.

In order to apply the tools that contain AI, methodologies have been adapted to suit the needs of the subjects. Two of them will be discussed and the choices that have been made as well as the reasoning behind them will be explained.

II. APPLICATION OF AI-POWERED SOFTWARE

A. Inclusion of AI Assistance in Analyzing Software Requirements and Specifications (ASRS)

At this point in time during their university education, software engineering students have already passed Databases (DBs), Object Oriented Programming (OOP), and some of their Algorithm subjects. They are introduced to the vision on how to lighten their workload using assistive tools powered by AI. ChatGPT is the text generation model that is currently being used, but as new AI chat bots emerge, university staff is testing them and showing students the differences between them.

In the subject ASRS, the students are required to submit a project – a software system that they must create and fully document using an iterative process. Our first use of ChatGPT is when we demonstrate its capabilities for idea generation.

As seen on Fig. 1, a description of the problem help is needed with is given on the left, and although it lacks detail, it is a true and valid statement, so the model gives a useful, albeit generic answer. On the right side the LLM is queried using a message that has more details included. It has been decided to include as much relevant information as possible as the specific requirements are narrowed down. A description of the field of study and level of progress in the field is defined – “student in Software engineering”, this can be further expanded by specifying that this is a bachelor's degree. By noting the current user skillset, the LLM can get software project ideas within the correct scope, omitting unfamiliar technologies. The

“course project” part of the message is also a component that will be used to further refine the scope of the ideas, giving them a timeframe. After defining what the LLM is required to do, the specified field is narrowed even further by statements of likes and dislikes. The overall structure of the message on the right helps the LLM print ideas that are more likely to interest the user and are within their means to achieve. The term “prompt engineering” [1] can better help describe this part of the communication with AI. It is highly possible that every person working with LLMs will be required to understand how to express themselves in a meaningful way if they want to be more efficient in the long term. Whether or not that will become a job requirement is speculative, but it is certainly a useful skill to have right now for students.

The logical path in narrowing down the choices that the model must make in its answer looks similar to this hierarchy in this case: field (of study or work) -> constraints (skillset, abilities, requirements) -> environment descriptions (question framework) -> task that has to be completed by the LLM (actual answer format) -> formatting (table, software code, bullet points, queries for other software) -> further requirements (inclusion and exclusion of topics and fields).

It is of course allowed for the students to use their own ideas irrespective of ChatGPT’s, and they are encouraged to

ask the AI for further fine-tuning – more functional requirements that will help their product become more robust, as well as help them expand their personal vision and make it better. The process is showcased step-by-step in real-time, but a video is prepared in case of service interruption during the demonstration. Outages in the OpenAI services have been less frequent recently, but during the peak hours they still happen, and it is important to have a good backup ready. Creating these videos is like a snapshot in time – it also helps showcase the differences in versions and their progress, as well as the time it takes to answer queries, model differences (3.5 and 4.0), as well as answer consistency. Recording the results has created data points for further research.

A showcase on how the model can play its role is also being created but requires more testing. Researchers have emulated an entire software development team creating and finishing a project [2]. The idea is to showcase how the different roles in a software development process are represented by AI (e.g., “Imagine you are a business analyst working on project 8 - Airport Logistics Planner. What would your workflow be? How would you approach every one of your team members? Please describe your activities using a table.”). Current focus is on the fact that users must highly customize their input and requirements gradually if they want to create and document a good and useful product.

You
Hello, please generate 10 ideas for a small software that has to be coded and documented. Put them in a table.

ChatGPT
Certainly! Here's a table with 10 software ideas along with a brief description for each:

Idea Number	Software Title	Description
1	Task Tracker	A simple task management tool with the ability to create, edit, and prioritize tasks with deadlines.
2	Expense Manager	An application to track and manage personal or business expenses, with features for budgeting and categorization.
3	Fitness Companion	A fitness app that allows users to log their workouts, track progress, and set fitness goals.
4	Recipe Organizer	An application for organizing and storing recipes, including features for meal planning and shopping lists.
5	Budget Analyzer	A financial tool that imports and analyzes bank statements to provide insights into spending habits and trends.
6	Language Learning Aid	A software to help users learn a new language, including vocabulary quizzes, pronunciation guides, and practice exercises.
7	Event Scheduler	An event management tool that enables users to schedule, coordinate, and manage events efficiently.
8	Habit Builder	An app focused on habit formation, allowing users to set goals, track daily habits, and receive reminders.
9	Mindfulness Assistant	A tool to support mindfulness and meditation practices, offering guided sessions and progress tracking.
10	Book Recommendation	A system that recommends books based on user preferences, reading history, and genre preferences.

These ideas cover a range of categories, from productivity and health to entertainment and personal development. Choose the one that aligns best with your interests and skills!

You
Assume I am currently a student in Software engineering. I already know how to work with databases, how to write code and interpret algorithms. I am currently brainstorming for ideas to write a course project on a topic of my choosing. I would like you to generate 10 ideas for a small software project that has to be coded and documented. Put them in a table. I want the ideas to be in a field of my liking - I like computer games, planes, trains, I hate cars and busses, as well as logistics. I also like Europe as a continent. Take my interests and the things I do not like into account when generating the ideas.

ChatGPT
Certainly! Considering your interests in computer games, planes, trains, and a dislike for cars and busses, as well as your fondness for Europe, here are 10 software project ideas tailored to your preferences:

Idea Number	Project Title	Description
1	Flight Simulator Companion	Develop a companion app for popular flight simulator games, offering in-flight checklists, weather updates, and navigation aids.
2	Train Timetable Tracker	Create a software tool to track and display train timetables across Europe, allowing users to plan and optimize their train travel.
3	EuroQuest: Geography Game	Design an educational game that quizzes users on European geography, landmarks, and capitals to make learning fun and interactive.
4	Virtual European Tour	Develop a virtual tour application that takes users on a journey through iconic European cities, showcasing historical landmarks and cultural hotspots.
5	Air Traffic Control Sim	Build an air traffic control simulation game where players manage air traffic at major European airports, emphasizing realistic scenarios and challenges.
6	Railway Tycoon Game	Create a strategy game where players can build and manage their own railway networks, dealing with challenges such as scheduling, maintenance, and competition.
7	European History Timeline	Develop an interactive timeline app that highlights key events in European history, providing users with a visually engaging way to explore historical periods.
8	Airport Logistics Planner	Design a logistics planning tool for airport operations, helping manage the flow of goods and resources efficiently within European airports.
9	Gaming Community Hub	Build a centralized platform for gamers interested in European-themed games, providing forums, reviews, and a curated list of games related to the region.
10	Rail Adventure Game	Create an adventure game set on a fictional train journey through Europe, incorporating puzzles, storytelling, and historical elements for an immersive experience.

Feel free to choose the one that resonates most with you or sparks your interest!

Fig. 1. Comparison between statements for idea generation, showcasing the influence of detailed information to the outcome.

The next step comes when the students have decided on their ideas for each team. At this point they have written most of the requirements, their vision, and they have mostly decided on what technological stack they will use. A showcase on how to create a relational DB that covers their functional requirements is done.

On Fig. 2 the process can be seen – starting with asking for the creation of several versions of the relations in a database and GPT must draw them with tables and connect them. When solutions are iterated several times, the first version of the database is required is completed. At this point SQL statements that will create the DB and its relations are required, specifying the type of software and SQL server they are going to run on. The demonstration currently uses MS SQL server through the Management Studio, but other clients are viable alternatives too – like PostgreSQL or MySQL. The demonstrations do not yet include experimenting with communicating with ChatGPT for NOSQL solutions, due to time constraints, but they are worth looking into.

You
Hello, I would like you to generate a simple database that can be used as a base for developing a small scale project for handling a restaurant, including its' menu and personnel's hours

ChatGPT
Creating a simple database for a restaurant management system involves designing tables to store information about menus and personnel hours. You can use a relational database management system like MySQL, PostgreSQL, or SQLite to implement this. Below is a basic schema for such a database:

Tables:

- Menu Items**
• This table stores information about the restaurant's menu items.

Column Name	Data Type	Description
item_id	INT	Primary Key
item_name	VARCHAR(50)	Name of the menu item
description	TEXT	Description of the menu item
price	DECIMAL(8,2)	Price of the menu item
category	VARCHAR(20)	Category (e.g. appetizer, main course)

- Personnel**
could you please generate the sql code I can run in management studio in order to create the database and its relations

ChatGPT
Certainly! Below is the SQL code to create the database and its related tables with relationships. You can run this code in SQL Server Management Studio (SSMS) or any other compatible database management tool:

```
sql
-- Create the RestaurantManagement database
CREATE DATABASE RestaurantManagement;
-- Use the RestaurantManagement database
USE RestaurantManagement;
-- Create the Menu Items table
CREATE TABLE MenuItems (
  item_id INT PRIMARY KEY,
```

Could you please generate data with insert statements that I can use to seed my database

ChatGPT
Certainly! Here's some sample data to insert into your restaurant management database tables using SQL INSERT statements. You can add these statements after creating your tables to seed your database with initial data:

```
sql
-- Insert data into the MenuItems table
INSERT INTO MenuItems (item_id, item_name, description, price, category)
VALUES
(1, 'Chicken Alfredo', 'Creamy chicken pasta', 15.99, 'Main Course'),
(2, 'Caesar Salad', 'Fresh romaine lettuce with Caesar dressing', 8.99, 'Appetizer'),
(3, 'Cheeseburger', 'Classic cheeseburger with fries', 10.49, 'Main Course'),
(4, 'Margherita Pizza', 'Thin crust pizza with tomato and cheese', 12.99, 'Main Course'),
(5, 'Chocolate Cake', 'Decadent chocolate cake', 6.99, 'Dessert');
-- Insert data into the Personnel table
INSERT INTO Personnel (staff_id, first_name, last_name, position, hire_date, hourly_rate)
```

Fig. 2. Asking ChatGPT with the creation of a database for a small-scale project.

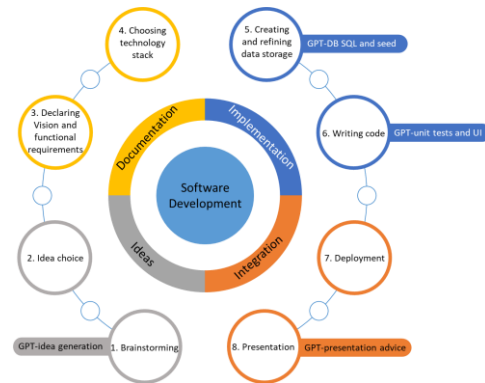


Fig. 3. Inclusion of AI in the software development process in the ASRS subject.

Another important step is showing how to “seed” the database with relevant data. First, a request that simply returns random raw data into several queries is created (see Fig. 3). Afterwards, if necessary, several key points and examples that the message should include in order to get relevant data are defined – for example random data in a special regular expression format is requested. Another option is to give GPT a short list with the example data, or the pattern to which it conforms. GPT can then work with it and iterate on it. This is followed by asking for the insert statements that are used to seed the finished database. There are cases where it has been noticed that there are differences in the format of the required data, as well as problems with relations – the foreign keys are not populated in the corresponding order, which leads to issues. Those issues are easily solvable for students that have passed their database-related subjects and further support the learning experience in working with such tools.

Students are shown how to ask for UI suggestions (mainly HTML and CSS), but as their chosen project tech stacks are different (games, desktop applications, web sites and mobile apps), the use of AI in the development part of the project is concluded here. It is important to note that there are interesting applications for creating unit tests that are mentioned. This part of the subject curriculum is still in development so at this stage students are only shown how to make simple unit tests for C# and how to base them on their functional requirements.

To put the finishing touch to their projects, hints are given on using the roleplaying ChatGPT to help them with their final presentations. A dialogue is shown following a similar structure to: “Hello, you are a business analyst. You are going to present a project on the topic of a flight fleet management software system. Your audience consists of people working in the sector. The project is made using the following technologies: C#, MSSQL, WebSocket, JavaScript and others. What would be your plan to present this project if you only have 15 minutes to talk?”. Students are usually very impressed, as most of them are used to using AI tools in other ways.

The use of AI powered software in the different steps of the software development process that is used in the ASRS subject can be seen in Fig. 3. As the focus is on creating requirements, specification and software documentation, it is acceptable to use AI for any of the other steps, as students have already learnt how to complete those by themselves. A strong

argument can be made that AI powered chatbots will generate the entire documentation themselves in the near future, but this stage of development is not yet reached. The guiding philosophy is that if students are able to finish a task by themselves and understand it well enough, they should be taught how to automate it at a future point in time.

B. Inclusion of AI Assistance in the Artificial Intelligence Subject

The use of assistive tools with AI in the subject "Artificial Intelligence" is presented in the second half of the semester. Traditionally the first half of the workshop tasks for the students consist of using Python to solve classic AI problems, such as eight queens and different crosswords, while using search and constraint satisfaction algorithms. After the introductory problems are taught, several more advanced topics and workshops are covered. Students are shown neural networks (NNs) - single layered and multilayered. They are tasked with creating several NNs on their own using the Keras library and MNIST data sets. The use of convoluted NNs in various fields such as in finances - predicting financial time series [4], gives the opportunity to teach students the interdisciplinary aspects of software engineering in general. After all, what use is software engineering if there is no field to attach software to? Giving working examples in actual businesses and practical applications gives confidence and generates ideas in our pupils.

In one of the following workshops, instructions on how to use open-source face recognition libraries in Python to detect faces on photos of famous people are included. There are training sets provided that are used to train and validate the model. In the researchers' experience, doing this workshop takes a significant amount of time – both the process to fulfill all the necessary requirements on each of their personal machines, and then follow the instructions and test the code they have written step by step are time-consuming. After the students have managed to go through all the steps and have created a working piece of software, they are encouraged to train and validate the model using their own set of photos as training sets.

Facial recognition, and image recognition in general, is not a new concept, but it is imperative students are taught how to apply it and use LLM tools on their own product to improve and test it. Students gain more thorough experience on how to modify their software products more efficiently by having access to the ChatGPT terminal, and thus to the entire collective training set using hundreds of billions of parameters that OpenAI have been using and improving.

A simple visual representation of the approach to teaching students in the AI subject is shown in Fig. 4.

The process of preparing additional workshops is constantly undergoing, and some have yet to be completed with students, as the subject is next semester in their curriculum. One workshop includes a comparison in image generation methods for different assisted tools, several of the most popular AI tools are presented - Stable Diffusion, Bing Images, Bard and Dall-E. A sufficiently difficult prompt using weight distribution for each of its parameters is used.

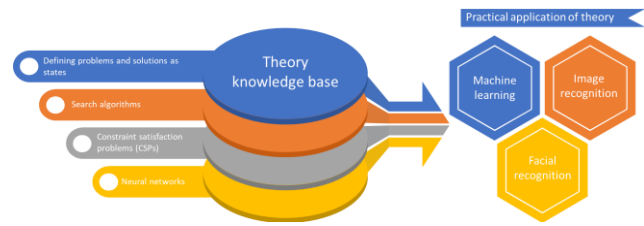


Fig. 4. Methodology for practical application of theoretical knowledge base in the education of students.

It is worth noting that Bard also hallucinated and tried to say it “drew” an image and responded with a google image search during one of the attempts to make it draw.

Fig. 5, 6, 7 and 8 display some of the collated results that were achieved using the currently most popular available AI-powered models. After multiple attempts, only Bing produced acceptable results, while Bard simply could not create any images itself, and returned googled images in almost all the attempts. The online version of DALL-E could not make images of better resolution and quality than those shown on Fig. 5. It can be said that testing web services in this way is not extensive enough, but when the services are unreliable themselves, produce different results and are in most cases limited, paid or of bad quality, they cannot be reliably used for educational purposes.



Fig. 5. Using Stable diffusion multiple times using a prompt.

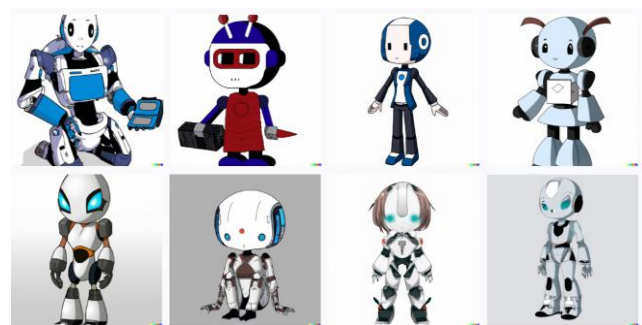


Fig. 6. Dall-E web using a prompt: “please draw a young robot that has a white coat, make the image ultra realistic”.

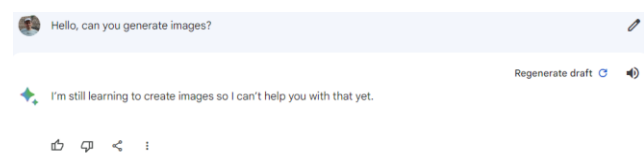


Fig. 7. Bard's response to drawing prompts.

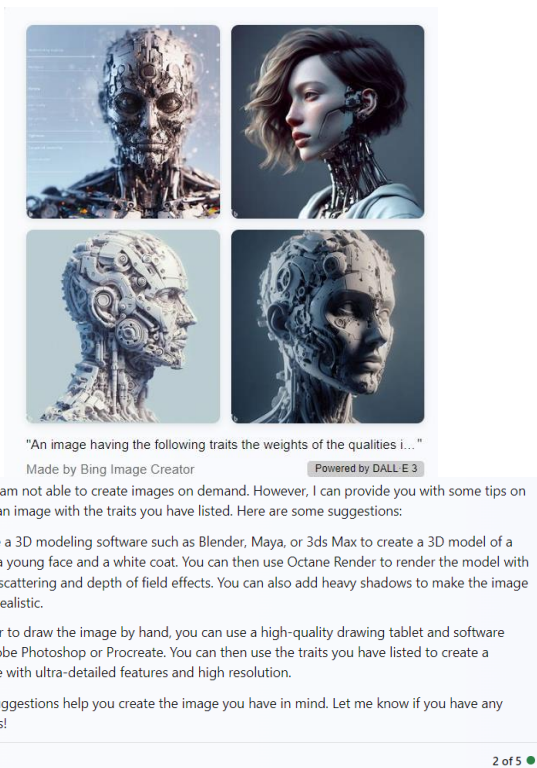


Fig. 8. Bing Images (Dall-E based).

It should be noted that Fig. 6 showcases the result of a very specific prompt with weight distribution that is as follows: “breathtaking, mysterious, fantasy, magical, (female robot:1.5) (creating a machine with hands), surrealism, hyper-realistic, colors and shapes, highly detailed, realism pushed to extreme, fine texture, 8k, ultra-detailed, (vivid swirling smoke, thick smoke:1.4), fluid, fire, cinematic, (intricate details:1.5), (vibrant colors:1.4), flash explosion, (colorful powder explosion), (billowing hair:1.5) “((magical enchantment on hands))” “((creates a machine from hands towards viewer))””. The LLM allows for incredibly detailed instructions that can be followed, such instructions are unavailable for the other tested tools currently. Fig. 5, 7 and 8 display the limitations of the most popular models. While spectacular results can also be achieved using them, it requires a bigger investment in prompting them, both in terms of time and money (as the paid model tokens are usually limited).

Some advantages and disadvantages are shown in Table I. The best results during the testing were achieved by StableDiffusion. All the other services had either service disruptions (Bing, Bard), hallucinations (Bing, Dall-E), required tokens and accounts (Bing, Dall-E), or simply didn't do as requested (Bing, Bard). The multitude of disadvantages make them unfit to be used in higher education. They are worth looking into further if their future iterations are improved. The currently chosen AI for the future workshops is StableDiffusion. Preliminary testing and scenarios based on the technology are being created and refined.

The methodology that has been adapted for an AI image generation workshop is shown on Fig. 9. It includes stable diffusion and showcases different models and their differences.

TABLE I. COMPARISON BETWEEN THE DIFFERENT AI POWERED IMAGE GENERATION TOOLS

AI	Advantages	Disadvantages	Notes
Bard	Free, on the web. Powered by Google.	Inconsistent behavior – sometimes says it cannot generate images; other times returns google image results.	Says it cannot generate images when prompted. After multiple attempts it starts to hallucinate that it can, in fact, generate images. The process itself is erroneous – it returns google image searches instead of generating images.
DALL-E	Free, on the web. Implemented in other software.	Low quality images, uses tokens, results are unsatisfactory.	Has bots in discord that work better than the software itself.
Bing	Free, uses Dall-e.	Only 5 questions. Has memory loss afterwards. Requires Microsoft Edge to work. Sometimes does not work – due to service disruption.	Says that it cannot generate images. Then generates images.
Stable Diffusion	Works offline. Can be downloaded for free. There are various community resources and models available. Active open-source project.	Requires hardware to run. Pre-trained models are at least several gigabytes each. Takes a long time to teach a model if hardware is not powerful enough.	While setting up the software can be difficult, the results are of great quality and require little manipulation before they can be used in production.



Fig. 9. Methodology for showcasing AI image generation to software engineering students.

Stable diffusion has an advantage for the current purposes - meaning it can be locally installed for free by anyone, and it doesn't require services or subscriptions. It does have hardware requirements, but they can be satisfied relatively easily for lower image resolutions. There is an enormous number of pre-trained models that are available, as well as learning resources from its community. Several different approaches to combining or altering images after their creation are discussed and prepared for presentation.

C. Introduction to Programming

For many students, the subject is the first time they work with algorithms and needing to understand what they are

doing. As it is taught during the first semester, the skill level difference is enormous. Preparing C++ tasks that are relevant and not demotivating is always a challenge everywhere in education [6, 7], some professors have very interesting solutions ways to solve the problem using games [8, 9]. The current established approach is to have automated tests for each small task we give the students, but it is still undecided on whether everyone should be introduced to testing at such an early semester. The benefits of test-driven development (TDD) are unquestionable [10, 11], but it requires a paradigm shift that is based on pre-existing knowledge in programming simple tasks, that many students lack during the first semester. Overloading fledgling software engineers with information is not the goal of the subject. The current idea is to use GPT to automate test creation process for the tasks that are given to the students for each workshop. The unit tests provide a good opportunity to limit test the code of students, but a reasonable way to enable them to access and understand them (as just giving a repository link is difficult for the average first-year student to understand) is not yet found. There are existing frameworks that provide a similar service [12], however they are not fit for this specific purpose.

D. Computer Graphics

There have been discussions on how exactly to integrate tools such as Stable Diffusion and creating applications using technology based on API that works with an AI that is locally deployed, but the complexity of the task is daunting for students at the average skill level during the fifth semester. What has been learned is that there needs to be an in-depth course that helps students familiarize themselves with the usage of image generating software on a deeper level. Software engineers should be able to not only create images – the way they are taught during the Artificial intelligence course, but also understand AI image generating software and how to write code for applications that use the tool themselves. They need to be able to alter the image generating algorithms of the AI models, and create models fit to their projects' needs.

E. Results and Observations

While the Dunning-Kruger effect has an impact on student self-assessment, research has shown that the differences between actual skill (and hence skill improvement) and self-evaluation are within acceptable levels [3]. It can be therefore proposed that students' self-assessment of their own skill improvement is significant enough to warrant noting a positive improvement due to the inclusion of AI in their curriculum subjects. Offloading the burden of menial tasks that can be easily automated leaves ample time for the actual improvement of their own professional interests and creating software products of value. One of the problems that have arisen is how professors evaluate whether the work done has led to improvement in a students' general skills, or just their skill in communicating with AI. That, however, is a complex enough problem to warrant research all by itself.

After questioning the students using a survey, as shown in Fig 10, it has been noticed that there is a good answer distribution among all the questions. It can be concluded that while it is useful, AI powered software needs improvement, and students using it need more experience to use it.

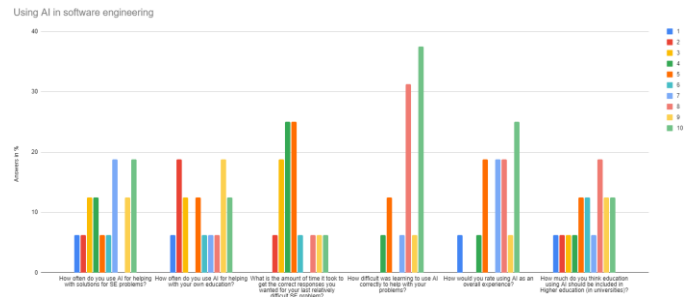


Fig. 10. Methodology for showcasing AI image generation to software engineering students.

What can be said with certainty is that students with the same university background have achieved success in projects that are more complex, more complete, and more in depth than those from previous years. According to the interviews with their teams during their project presentations, AI has certainly played a significant role in “doing the heavy lifting” for most of the teams. The most noticeable improvement is in the lower achieving students – given such powerful tools, they can elevate their skill level to a degree, sufficient for them to be motivated enough to be able to complete a reasonably sized software project. Compared to previous semesters in the same subject - students on the low skill spectrum could not finish as much work in the same amount of time. Elevating the results less skilled learners can achieve is motivating for professors as well. This gives confidence, experience, and motivation to the students, which in turn helps with student retention and an improvement to the system in higher education. Lifting the floor is beneficial to improving educational levels [5].

III. CONCLUSION

Software engineering students need more exposure to AI tools in higher education. We need to have specialized subjects to help them familiarize themselves with this type of technology. It is imperative that they are not placed in a position where they have to “make do” but have the opportunity to learn how to leverage AI in their projects.

It is important to note that all LLMs often hallucinate, and because of that they are not reliable for checking facts and truth. At this point of their evolution, it is difficult to use them in the same way search engines are trustworthy data sources are used. What they are incredible at though, is giving ideas, workflows, helping with automation and doing the heavy lifting when beginning and polishing projects. This is what we are trying to teach our students, that they must be aware of the strengths and weaknesses of tools that apply AI.

With any emerging nascent technology, there is more testing required in order to make it work better. Familiarizing students with the correct approach to using AI assistive tools - focusing on their strengths and being aware of their weaknesses, is imperative for them to not be in a disadvantageous position in their future careers. As AI grows, everyone not aware of how to take advantage of it will quickly become a less valuable employee for the business. Starting too early is detrimental to brain development and problem-solving skills, but starting late is detrimental to career opportunities and life quality. Communication skills will become even more

important in the future but acquiring them will require more effort with the further digitization of communication and the lessening of social relationships people experience.

ACKNOWLEDGMENT

This research is supported by the Bulgarian Ministry of Education and Science under the National Program “Young scientists and Postdoctoral Students – 2.

REFERENCES

- [1] Oppenlaender, Jonas. "Prompt engineering for text-based generative art." arXiv preprint arXiv:2204.13988 (2022).
- [2] Qian, Chen, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. "Communicative agents for software development." arXiv preprint arXiv:2307.07924 (2023).
- [3] Gignac, Gilles E., and Marcin Zajenkowski. "The Dunning-Kruger effect is (mostly) a statistical artefact: Valid approaches to testing the hypothesis with individual differences data." *Intelligence* 80 (2020): 101449.
- [4] Markova, Maya. "Convolutional neural networks for forex time series forecasting." In *AIP Conference Proceedings*, vol. 2459, no. 1. AIP Publishing, 2022.
- [5] Crouch, Luis, and Caine Rolleston. "Raising the floor on learning levels: Equitable improvement starts with the tail." *RISE Insights* 2016 (2017): 1-14.
- [6] Aung, Shune Lae, Nem Khan Dim, Soe Mya Mya Aye, Nobuo Funabiki, and Htoo Htoo Sandi Kyaw. "Investigation of Value Trace Problem for C++ Programming Self-study of Novice Students." *International Journal of Information and Education Technology* 12, no. 7 (2022): 631-636.
- [7] Alzahrani, Nabeel, Frank Vahid, Alex Edgcomb, Kevin Nguyen, and Roman Lysecky. "Python Versus C++ An Analysis of Student Struggle on Small Coding Exercises in Introductory Programming Courses." In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, pp. 86-91. 2018.
- [8] Ariffin, Mazeyanti Mohd, Nurshazlyn Mohd Aszemi, and Mohammad Syazran Mazlan. "CodeToProtect@: C++ programming language video game for teaching higher education learners." In *Journal of Physics: Conference Series*, vol. 1874, no. 1, p. 012064. IOP Publishing, 2021.
- [9] Agapito, Jenilyn L., Joshua C. Martinez, and J. D. Casano. "Xiphias: A competitive classroom control system to facilitate the gamification of academic evaluation of novice C++ programmers." In *Proceedings of International Symposium on Computing for Education, ISCE*, vol. 14, pp. 9-15. 2014.
- [10] Beck, Kent. *Test driven development: By example*. Addison-Wesley Professional, 2022.
- [11] Langr, Jeff. "Modern C++ Programming with Test-driven Development: Code Better, Sleep Better." *Modern C++ Programming with Test-Driven Development* (2013): 1-368.
- [12] Markoska, Ramona. "Managing ICT solutions for training and evaluation of C++ programming skills in e-learning ecosystem." *New Trends and Issues Proceedings on Humanities and Social Sciences* 6, no. 7 (2019): 33-41.